Desain dan Implementasi Convolutional Encoder (2, 1, 8) dalam Field Programmable Gate Array (FPGA)

Ferry Wahyu Wibowo
Jurusan Teknik Informatika, STMIK AMIKOM Yogyakarta
Jl. Ring Road Utara, Depok, Sleman, Yogyakarta, telp. 08157948404
e-mail: ferrywahyu@gmail.com

***Abstrak—Convolutional encoder* sudah lama digunakan dalam dunia sistem komunikasi digital, bahkan sistem komunikasi digital modern masih menggunakan *encoder* tersebut. Prinsip *convolutional encoder* merupakan proses yang digunakan untuk menambah redundansi pada aliran sinyal yang masuk. Perancangan *convolutional encoder* banyak yang menggunakan rancangan *finite state machine* dan sudah berupa *intelectual property (IP) core*. *IP core* tersebut sangat membantu perancang perangkat keras jaringan dalam menganalisa dan merancang aplikasi sistem jaringan dalam pengiriman dan penerimaan data. Paper ini memaparkan implementasi *field programmable gate array (FPGA)* untuk membuat *convolutional encoder*. Prinsip yang digunakan dalam paper ini adalah merancang *convolutional encoder* menggunakan komponen pada tingkat *register transfer logic (RTL)*. Komponen yang digunakan dalam rancangan adalah register geser (*shift register*), penjumlah (adder) dan *mutiplekser*. Hasil sintesis yang diperoleh dalam FPGA Spartan-3E untuk mengimplementasikan *convolutional encoder* (2, 1, 8) mengkonsumsi komponen 3 *slices*, 6 *slice flip-flop*, 4 4-*input LUT*, 4 IO, 4 *bonded IOB* dan 1 *GCLK*, sedangkan waktu yang diperlukan untuk memproses data masukan pada *convolutional encoder* berbasis *FPGA* sebesar 6,07ns.**

Kata Kunci. ***FPGA, convolutional encoder, komunikasi, digital, RTL.***

I. INTRODUKSI

Sistem komunikasi digital modern dirancang untuk mengirimkan data yang sangat besar secara sekuensial. Data yang dikirimkan berupa sinyal serial yang berurutan. Pengiriman sinyal data menggunakan prinsip nirkabel sangat rentan untuk berinterferensi dan hilang dalam proses perjalanannya menuju perangkat penerima (*receiver*). Sinyal data yang diolah oleh *modulator* akan diubah menjadi sinyal analog yang berupa sinyal frekuensi radio. Sinyal frekuensi radio tersebut akan diterima oleh perangkat penerima dan akan diteruskan ke *demodulator*, sehingga paket data yang dikirimkan tersebut akan diolah agar dapat dibaca dan diterjemahkan untuk difungsikan lebih lanjut. Salah satu teknik yang digunakan dalam aplikasi *modulator* dan *demodulator* untuk pengolahan sinyal data adalah teknik *convolutional code*. Teknik *convolutional code* telah digunakan sejak tahun 1955 di dalam sistem teknologi telekomunikasi di berbagai macam aplikasi seperti CDMA, *voiceband modem* dan komunikasi satelit [1]. *Convolutional code* sering digunakan sebagai *error correction code*. *Error correction code* (ECC) merupakan redundansi pada aliran bit informasi agar mengijinkan deteksi dan koreksi *error* simbol saat transmisi [2]. ECC dapat digunakan untuk komunikasi yang *real-time* dengan informasi yang berurutan pada suatu aliran sinyal data menggunakan penambahan derau, *bit error* ataupun distorsi sinyal yang ditransmisikan [3], sehingga sinyal data yang ditransmisikan tersebut dapat terjaga keintegritasan datanya ketika ditangkap oleh piranti penerima. Proses *decoding* resultan aliran data pada penerimaan *node* dapat mempunyai proses yang sangat kompleks, sedangkan implementasi sistemnya sangat sederhana. *Convolutional encoding* dari suatu data dikombinasikan dengan Viterbi *decoding* pada penerimaan node telah banyak digunakan oleh standar industri untuk saluran digital [4]. *Convolutional encoding* menunjukkan kinerja *forward error correction* (FEC) dan mempunyai peningkatan yang cukup signifikan dalam teknologi proses *very large scale integration* (VLSI). Teknologi VLSI memungkinkan untuk merealisasikan *encoder/decoder chip* yang mempunyai kecepatan yang tinggi. *Convolutional code* beroperasi pada data serial dalam sekali waktu [5]. *Convolutional code* dengan rangkaian tunda tambahan dapat digunakan sebagai *burst-error correction* [6]. Namun, masalah yang dihadapi para peneliti adalah menguji hasil eksperimental atau simulasi suatu klas *code* [7].

Penelitian dengan membandingkan penggunaan *convolutional encoder* dengan teknik yang lain telah dilakukan oleh beberapa penelitian. Asif, S. M. et. al. [8] mengamati bahwa *convolutional code* lebih baik dalam menangani *bit error rate* (BER) daripada sinyal yang tidak dikodekan pada saluran *Rayleigh fading* lambat. Paper Ebian et. al. [9,10] menjelaskan bahwa *convolutional product code* (CPC) lebih baik dalam menangani *signal-to-noise-ratio* (SNR) yang berbeda dalam aplikasi *Worldwide Interoperability for Microwave Access* (Wi-MAX) daripada menggunakan *turbo code*. Mereka mengamati pengaruhnya pada modulasi yang berbeda, yaitu 16QAM – 64QAM dan jumlah OFDMA *sub-carrier* yang berbeda (128-512). Juleen dan Kiong [11] meneliti *convolutional encoder* (2, 1, 8) yang diaplikasikan dalam bahasa pemrograman C++ dan mereka mengajukan saran agar *convolutional encoder* diimplementasikan dalam perangkat keras (*hardware*), karena implementasinya mempunyai keamanan yang lebih kuat daripada diimplementasikan berbasis perangkat lunak (*software*). Patil, et. al. [12] telah meneliti dengan mengaplikasikan *convolutional encoder* menggunakan *hardware description language* (HDL), rancangan yang diajukan menggunakan prinsip *finite state machine* (FSM). Perancangan menggunakan HDL dapat

mempunyai model sintesis konsumsi komponen FPGA yang sama untuk beberapa teknik perancangan [13].

Paper ini memaparkan *convolutional encoding* yang diimplementasikan dalam *field programmable gate arrays* (FPGAs) pada tingkat *register transfer logic* (RTL) sebagai platform *system-on-chip* (SoC). SoC merupakan tren masa depan yang dapat diandalkan karena implementasi suatu algoritma dapat ditanamkan pada sebuah *chip* dengan area dan kecepatan kinerja yang mempunyai *trade-off* [14]. Desain yang dibuat menggunakan komponen-komponen dasar sistem digital yang ditulis dalam kode *very high speed integrated circuit hardware description language* (VHDL). Paper ini juga membandingkan antara *convolutional encoder* yang telah dirancang dengan *convolutional encoder* yang telah diaplikasikan dalam bentuk *intellectual property* (IP) *core* menggunakan FPGA jenis lain, tujuannya untuk mengetahui tingkat efektivitas dan efisiensi rancangan di dalam implementasi FPGA. Hasil yang diperoleh dalam paper ini adalah rancangan, konsumsi komponen yang digunakan untuk mengimplementasikan *convolutional encoder* (2, 1, 8) dan waktu perambatan pengolahan sinyal untuk memproses data masukan di dalam FPGA.

II. LANDASAN TEORI

A. *Convolutional Code*

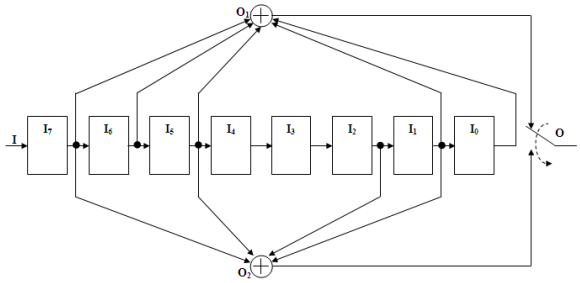
Convolutional code merupakan jenis dari *error correcting codes* (ECCs), sehingga kode ini sering digunakan untuk melindungi sistem dari *error* pada saat pengkodean serangkaian informasi di saluran penerima [15]. Serangkaian informasi dibagi menjadi blok kecil sejumlah masukan k dan dienkode menjadi simbol sejumlah keluaran n. *Convolutional encoder* (n, k, m) diimplementasikan sebagai serangkaian sekuensial linear dengan sejumlah masukan k, keluaran n dan urutan memori m. Biasanya nilai n dan k merupakan bilangan integer yang bernilai kecil, nilai k lebih besar daripada nilai n (k>n), tetapi nilai m lebih besar daripada nilai k. Serangkaian informasi, jika tidak dibagi dalam blok dan nilai k=1, maka serangkaian informasi tersebut dapat diproses secara langsung.

Convolutional encoder (n, k, m) menandakan nilai R. Nilai R merupakan nilai masukan k dibagi dengan nilai keluaran n, dituliskan sebagai R=k/n, dan mempunyai tahap *encoder* dengan urutan memori m sama dengan nilai batasan kode K dikurangi 1, dituliskan sebagai m=K-1. Urutan atau panjang nilai memori *encoder* m sama dengan panjang tunda serangkaian data tersebut. Misalkan, *convolutional encoder* (2, 1, 8) mengandung makna bahwa kode tersebut mempunyai nilai R=1/2 dan nilai memori m=8, sehingga nilai panjang batas K=9 [11]. Persamaan dari prinsip *convolutional encoding* yang mempunyai dua keluaran, O_1 dan O_2, dengan *generator polynomial* 343_8 dan 246_8, mempunyai arti bahwa nilai yang diambil dari keluaran O_1 berada pada data keluaran register I_7, I_6, I_5, I_1 dan I_0, sedangkan keluaran O_2 berada pada keluaran register I_7, I_5, I_2 dan I_1. Persamaan O_1 dan O_2 ditunjukkan menurut persamaan (1) dan persamaan (2).

$$O_1 = I_7 + I_6 + I_5 + I_1 + I_0 \quad (1)$$

$$O_2 = I_7 + I_5 + I_2 + I_1 \quad (2)$$

Bagan *convolutional encoder* dari prinsip persamaan (1) dan (2) ditunjukkan menurut Gambar 1.



Gambar 1. Bagan *convolutional encoder*

Penelitian Viraktamath et. al. [5] menunjukkan langkah-langkah untuk mengenkode data menggunakan *convolutional encoder*. Langkah-langkah enkode data tersebut dapat dijadikan sebagai acuan untuk mengimplementasikan FPGA sebagai *convolutional encoder* dengan mengambil *generator polynomial* yang sesuai dengan jumlah register geser (*shift register*) yang diaplikasikan dalam *encoder*. Data masukan yang diperoleh kemudian ditransmisikan dengan mencari panjang *generator polynomial* dan panjang data. Data masukan kemudian digeser satu bit data ke *shifter* sesuai dengan masukan *clock*, sedangkan enkode data menggunakan *generator polynomial* yang diberikan. Setelah semua data masukan telah dienkode, maka proses *encoding* dihentikan.

B. *Field Programmable Gate Array (FPGA))*

FPGA dapat diimplementasikan untuk proses komputasional yang menggunakan unsur aritmatika, yang terdiri dari perkalian dan penjumlahan dengan mengurangi aspek pemrograman perangkat lunak dan mempunyai faktor *error* yang kecil [16]. FPGA merupakan piranti yang terdiri dari jutaan gerbang dan dapat dikonfigurasi dengan memasukkan berbagai algoritma dan fungsi tertentu untuk mendapatkan suatu modul perangkat keras yang mengacu pada sistem digital. FPGA dapat dikonfigurasi menggunakan 3 cara, yaitu skematik, *state-diagram*, dan bahasa deskripsi. Perancangan menggunakan teknik skematik hanya memilih dan meletakkan komponen atau modul yang telah disediakan oleh perangkat lunak. Modul tersebut bisa dirangkai lebih besar dengan modul yang lain sehingga didapatkan modul yang komprehensif. Perancangan skematik menggunakan dasar gambar untuk membuat rancangannya. Perancangan *state-diagram* menggunakan dasar gambar juga, namun rancangan ini didasarkan pada alur yang mempunyai prinsip *finite state machine* (FSM). Perancang perangkat keras yang memahami aspek sekuensial suatu rancangan dapat mengaplikasikan teknik ini, karena hanya menggunakan gambar simbol proses dan panah saja. Namun untuk membuat suatu rancangan yang tidak didasarkan pada suatu proses, maka akan sangat sulit untuk mengaplikasikannya. Perancangan menggunakan bahasa deskripsi lebih fleksibel. Saat ini ada dua macam bahasa deskripsi yang sering digunakan, meskipun ada bahasa deskripsi yang lain, yaitu Verilog dan VHDL. Penelitian ini menggunakan bahasa deskripsi VHDL. VHDL kepanjangan dari *Very High Speed Integrated Circuit High Description Language*, yang digunakan untuk pemodelan sistem digital dan sebagai metodologi perancangan *top-down* [17]. Eksekusi kode VHDL disebut sebagai sintesis, yang menghasilkan *netlist*. *Netlist* tersebut berisi daftar komponen yang digunakan dalam perancangan, dan dari *netlist* tersebut dapat ditentukan konsumsi FPGA.

Kemampuan FPGA yang bersifat dapat dikonfigurasi ulang, membuat piranti ini sangat sesuai dalam platform perancangan perangkat keras untuk mengetahui tingkat efektivitas dan efisiensi suatu rancangan. Algoritma rancangan dan pemahaman dalam bahasa deskripsi perangkat keras sangat menentukan dalam proses pembuatan rancangan modul. Perancangan berbasis FPGA secara garis besar dibagi menjadi dua model. Model perancangan *behavioral* menggunakan prinsip algoritma abstraksi yang hanya mementingkan watak dari rancangan yang hendak dibuat, sehingga terkadang seorang perancang perangkat keras tidak perlu memahami dasar-dasar sistem digital. Model perancangan struktural menggunakan prinsip modular, sehingga seorang perancang perangkat keras menggunakan prinsip dasar-dasar sistem digital untuk membuat rancangannya. Namun jarang sekali ditemukan pemisahan antara model perancangan *behavioral* dan model struktural. Dalam perancangan perangkat keras, kedua model perancangan saling terkait satu sama lain.

FPGA yang digunakan dalam penelitian ini adalah Xilinx Spartan-3E, sedangkan perangkat lunak yang digunakan untuk memprogram FPGA adalah Xilinx ISE 9.2i. FPGA Xilinx Spartan-3E terdiri dari *component logic blocks* (CLBs) yang tersusun dari beberapa *slice*. Satu slice terdiri dari dua *look-up-table* (LUT) dan *flip-flop* (FF). Penghubungan antar *slice* satu dengan yang lain diatur oleh *switch matrix*.

III. METODE PENELITIAN

Eksperimental yang dilakukan dalam penelitian ini, mengimplementasikan FPGA Spartan 3E *starter kit* yang digunakan untuk membuat *convolutional encoder* (2, 1, 8). Langkah-langkah yang dilakukan untuk mengkonfigurasi FPGA terdiri dari:

a. Spesifikasi, merupakan tahap perancangan untuk membuat komponen yang diperlukan dalam perancangan *convolutional encoder* (2, 1, 8) menggunakan VHDL pada perangkat lunak Xilinx ISE 9.2i. Hasil yang didapatkan berupa sintesis *register transfer logic* (RTL) yang diterjemahkan menjadi *netlist* dari LUT dan FF.

b. Verifikasi, merupakan tahap simulasi sintesis dari kode VHDL pada rancangan *convolutional encoder* (2, 1, 8). Hasil yang didapatkan berupa diagram bentuk gelombang masukan dan keluaran pada FPGA Spartan-3E. Tahap ini juga digunakan untuk menentukan waktu tunda yang dihasilkan oleh komponen logika dan penjalarannya dalam memproses sinyal data masukannya.

c. Implementasi, merupakan tahap penterjemahan, pemetaan, dan penyambungan komponen yang dihasilkan oleh *netlist* untuk diprogramkan ke dalam FPGA Spartan-3E.

d. Debug dan pengujian, merupakan tahap validasi rancangan yang diaplikasikan dalam FPGA dengan menggunakan fitur-fiturnya. Kesesuaian dan integritas antara tahap verifikasi dan implementasi dapat dipantau dari tahap pengujian ini.

Implementasi FPGA dalam perancangan perangkat keras *convolutional encoder* (2, 1, 8) menggunakan prinsip tingkat *register transfer logic* (RTL). Sistem yang dibangun pada tingkat RTL didasarkan dari sistem komponen digital, yang terdiri dari register geser (*shift register*), penjumlah (*adder*) dan multiplekser. Metode yang dibuat berdasarkan pada metode *top-down*. Metode top-down merupakan metode dengan mendeskripsikan modul secara keseluruhan terlebih dahulu sebelum membagi modul ke dalam sub-sub modul, sehingga didapatkan suatu rancangan yang lebih spesifik.

IV. HASIL DAN PEMBAHASAN

A. Hasil Perancangan

Desain *shift register* yang digunakan dalam paper ini menggunakan *register* DFF satu bit yang terangkai seri sebanyak 8 buah. Masing-masing *register* DFF dikendalikan oleh *clock* untuk mengendalikan penjalaran datanya. *Generator Polynomial* digunakan untuk menentukan hubungan keluaran dari masing-masing *register* DFF dengan keluaran *encoder*. Rangkaian penjumlah bit-bit dari keluaran *register* yang formatnya sesuai dengan *generator polynomial* untuk desain *convolutional encoder* adalah rangkaian XOR bukan rangkaian penjumlah (*adder*), karena *carry* tidak signifikan kedudukannya di dalam perancangan rangkaian *convolutional encoder*. Sintesis rancangan rangkaian XOR antara dari vendor Xilinx dan hasil perancangan menggunakan gerbang-gerbang dasar yang dibuat dari kode VHDL mempunyai struktur dan konsumsi komponen yang sama dalam FPGA. Rangkaian XOR hanya menggunakan kombinasi tiga gerbang dasar, yaitu gerbang NOT, AND dan OR, yang disusun sebagai *((NOT(A) AND B) OR (A AND NOT(B))*.

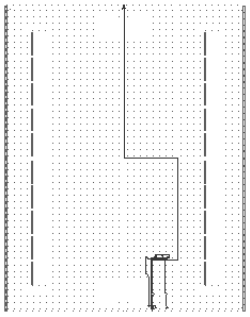
Konsumsi komponen FPGA berperan sangat penting dalam menyusun komponen-komponen dasar untuk membuat suatu rangkaian *convolutional encoder* yang efektif. Konsumsi komponen tersebut mempunyai dampak yang menentukan tingkat *trade-off* antara area dan kecepatan dari desain yang dibuat. Hasil sintesis digunakan untuk melaporkan hasil analisa perancangan komponen pembentuk rangkaian dalam FPGA. Sintesis yang menunjukkan konsumsi komponen rangkaian *convolutional encoder* (2, 1, 8) pada FPGA Spartan-3E ditunjukkan pada Tabel I.

TABEL I
KONSUMSI KOMPONEN FPGA YANG DIPERGUNAKAN UNTUK MEMBUAT *CONVOLUTIONAL ENCODER* (2, 1, 8)

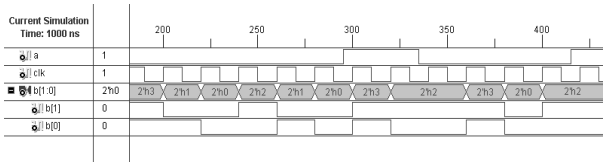
No.	Komponen	Konsumsi
1	*Slices*	3 dari 4656 (0%)
2	*Slice Flip-Flop*	6 dari 9312 (0%)
3	4 Masukan LUT	4 dari 9312 (0%)
4	IO	4
5	IOB	4 dari 232 (1%)
6	GCLK	1 dari 24 (4%)

TABEL II.
KONSUMSI KOMPONEN FPGA ACTEL YANG DIGUNAKAN UNTUK MEMBUAT *CONVOLUTIONAL ENCODER*

No.	FPGA Actel	Piranti	R/C-cell	Konsumsi
1	ProASICPlus	APA075-STD		26%
2	Axcelerator	AX125-3	31/29	3%
3	RTSX-S	RT54SX32S-1	31/29	3%
4	SX/SX-A	54SX08A-1	31/29	8%



Gambar 2. *Floorplan* dari *convolutional encoder*



Gambar 3. Diagram bentuk gelombang dari *convolutional encoder*

Konsumsi komponen dari FPGA Spartan-3E yang ditampilkan pada Tabel I menunjukkan bahwa *configurable logic blocks* (CLBs) yang digunakan untuk mendesain *convolutional encoder* (2, 1, 8) sangat kecil dari sumber yang tersedia. Perbandingan perancangan dalam konsumsi komponen *convolutional encoder* yang didapatkan dari FPGA Actel mempunyai asumsi konsumsi komponen yang lebih besar daripada perancangan dalam penelitian ini, yaitu dengan mengimplementasikan *convolutional encoder* dalam bentuk *core* yang mempunyai fitur panjang K=7, dan menggunakan *generator polynomial* $g_0=171_8$, $g_1=165_8$ (R=1/2 atau 1/3), dan $g_2=133_8$ (R=1/3), [18] (lihat Tabel II).

B. Hasil Implementasi

Gambar *floorplan* dari perancangan *convolutional encoder* (2, 1, 8) terdiri dari konsumsi komponen yang digunakan oleh FPGA Spartan 3E, sebagaimana ditunjukkan pada Gambar 2.

Area FPGA yang dipakai untuk merancang *convolutional encoder* (2, 1, 8) mengkonsumsi komponen yang sangat kecil sekali dibandingkan dengan luas area yang tersedia dalam FPGA, diperlihatkan pada gambar 2 dengan garis agak tebal yang berada di tengah gambar. Pada gambar 2 juga terlihat garis panjang yang menjalar, garis tersebut merupakan penyambungan dengan *global clock* GCLK. Pada dasarnya penempatan dan penyambungan antar komponen dapat diatur sesuai dengan penggunaan pin-pin yang difungsikan sebagai masukan dan keluaran pada FPGA.

Periode minimum yang dihasilkan dari implementasi *convolutional encoder* (2, 1, 8) pada FPGA Spartan-3E sebesar 4,014ns atau dengan kata lain mempunyai frekuensi maksimum sebesar 249,128MHz. Periode minimum merupakan perkiraan periode *clock* yang digunakan oleh komponen sesudah disintesis, tepatnya sesudah *placement and routing*, sehingga dapat digunakan untuk menentukan waktu tunda terburuk dalam penggunaan *clock* antar komponen. Waktu yang dihasilkan oleh sintesis untuk rangkaian kombinasional pada desain rangkaian *convolutional encoder* (2, 1, 8) sebesar 6,07ns yang terdiri dari 4,888ns *logic* (80,5%) dan 1,182ns *route* (19,5%). Simulasi pewaktuan rangkaian *convolutional encoder* antara sinyal masukan dan keluarannya yang telah diimplementasikan dalam FPGA Spartan-3E digambarkan melalui diagram bentuk gelombang yang ditunjukkan pada Gambar 3.

Proses pengolahan data pada *convolutional code* (2, 1, 8) yang diimplementasikan dalam FPGA mempunyai tingkat akurasi yang tinggi dan lebih cepat dalam pemrosesan datanya dibandingkan dengan mengimplementasikan *convolutional code* berbasis perangkat lunak, sebagaimana yang telah ditunjukkan oleh [11]. Implementasi *convolutional encoder* dalam perangkat keras akan sangat baik jika digunakan dalam dunia telekomunikasi, terutama dalam aplikasi yang lebih kompleks untuk mendapatkan efisiensi dan efektivitas dalam pensinyalan datanya. Walaupun demikian, implementasi suatu piranti harus mempertimbangkan tingkat fungsi, kegunaan dan kebutuhan dari perancangan perangkat keras. FPGA untuk saat ini memang sangat mahal, namun sangat efektif untuk mendesain suatu perangkat keras atau hanya digunakan sebagai *softcore* untuk membentuk *core* yang lebih luas lagi.

V. KESIMPULAN

Paper ini menunjukkan desain rancangan *convolutional encoder* yang dibuat menggunakan teknologi *very large scale integration* (VLSI) berbasis FPGA yang diprogram menggunakan kode VHDL. Desain *convolutional encoder* sangat sederhana, karena hanya menggunakan beberapa *shift register* dan beberapa rangkaian XOR, namun dapat digunakan sebagai deteksi dan koreksi *error* yang efektif. Hasil sintesis dari FPGA Spartan-3E menunjukkan bahwa konsumsi *slice* yang digunakan dalam perancangan *convolutional encoder* (2, 1, 8) hanya sebesar 3 *slices* dari 4656 *slices* yang tersedia atau sekitar 0% saja, sehingga masih memungkinkan untuk dikembangkan dan dikombinasikan dengan desain lain untuk membentuk desain yang lebih kompleks. Hasil perancangan menggunakan FPGA Xilinx lebih efisien daripada menggunakan FPGA Actel. Namun, hasilnya akan lebih efektif dan efisien lagi jika menggunakan FPGA yang telah terintegrasi dengan *digital signal processing* (DSP).

DAFTAR PUSTAKA

[1] Rhee, M. Y. *CDMA Cellular Mobile Communications Network Security*. Prentice Hall PTR: Prentice-Hall, Inc, 1998.

[2] Aadil, F., Nisar, S. K., Abbas, W., Shahzad, A. Reusable IP Core for Forward Error Correcting Codes. *International Journal of Basic & Applied Sciences IJBAS-IJENS, 10*(01), pp. 24-30, 2010.

[3] Wessel, R. D. Convolutional Code. *Wiley Encyclopedia of Telecommunications*, pp. 1-8, 2003.

[4] Singh, B., Agarwal, S., Varma, T. Hardware Implementation of Viterbi Decoder for Wireless Applications. *Proceedings of the International Joint Journal Conference on Engineering and Technology (IJJCET 2010)*, 2010.

[5] Viraktamath, S. V., Attimarad, G. V. Impact of Constraint Length on Performance of Convolutional CODEC in AWGN Channel for Image Application. *International Journal of Engineering Science and Technology, 2*(9), pp. 4696-4700, 2010.

[6] Kong, J. J. and Parhi, K. K. Interleaved Convolutional Code and Its Viterbi Decoder Architecture. *EURASIP Journal on Applied Signal Processing, 13*, pp. 1328-1334, 2003.

[7] Ouahada, K., Ferreira, H.C. On the Generating Function and Its Application to Ternary Line Codes. *South African Institute of Electrical Engineers, 101*(4), pp. 121-131, 2010.

[8] Asif, S. M., Al-Maruf, A., Islam, M. A., Tikader, A., and Alim, M. A. Comparison of BER between Uncoded Signal and Coded Signal (using Convolution Code) Over Slow Rayleigh Fadding Channel. *Journal of Theoretical and Applied Information Technology*, pp. 76-82, 2005.

[9] Ebian, A., Shokair, M., and Awadalla, K. H. Performance Evaluation of WIMAX System using Convolutional Product Code (CPC). *Progress In Electromagnetics Research C, 5*, pp. 125-133, 2008.

[10] Ebian, A., Shokair, M., and Awadalla, K. H. Comparison between Turbo Code and Convolutional Product Code (CPC) for Mobile WiMAX. *World Academy of Science, Engineering and Technology, 51*, pp. 195-199, 2009.

[11] Juleen, L., Kiong, T. S. Dynamic (2,1,8) Convolutional Coding Algorithm for Data Correction and Security Enhancement in an Intelligent Urban Traffic Management System. *ICGST-CNIR Journal, 5*(3), pp. 29-34, 2006.

[12] Patil, G. U., Nirmal, N. G., Chaudhari, P. P. Implementation of Convolutional Encoder in HDL. *International Journal of Advanced Engineering and Application*, pp. 22-25, 2011.

[13] Wibowo, F. W. The Conservative Structure of Synthesizing Read Only Memory Design using VHDL on FPGA. *International Seminar on Industrial Engineering and Management, 4*, pp. 231-235. 2010.

[14] Wibowo, F. W. System on Chip untuk Mesin Pengepakan Barang Berbasis FPGA, *Seminar Teknik Informatika 2011*, pp. B-52-B-59, 2011.

[15] Azim, C. F., Dur-e-Jabeen, Ahmed, S.F. FEC Performance Enhancement of Mobile WIMAX by Controlling Trace Back Length. *International Journal of Academic Research, 3*(1), pp. 41-46, 2011.

[16] Agrahari, A., Darji, A. FPGA Implementation of Baseband Modulation for Transceiver of 802.16e (Wi-MAX). *International Journal of Engineering Science and Technology, 2*(8), pp. 3721-3728, 2010.

[17] Wibowo, F. W. Implementasi FPGA untuk Pengukuran Daya Listrik. *Seminar Nasional Fisika 2010*, pp. FI106-1 – FI106-6, 2010.

[18] Actel. Convolutional Encoder and Viterbi Decoder Core Datasheet For Actel FPGAs. Actel Companion Core, 4i2i Communications Ltd.